



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/726,902	12/03/2003	Mitchell Alsup	5500-88700	4177

53806 7590 08/23/2006

MEYERTONS, HOOD, KIVLIN, KOWERT & GOETZEL (AMD)
P.O. BOX 398
AUSTIN, TX 78767-0398

EXAMINER

FENNEMA, ROBERT E

ART UNIT PAPER NUMBER

2183

DATE MAILED: 08/23/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No. 10/726,902	Applicant(s) ALSUP ET AL.	
	Examiner Robert E. Fennema	Art Unit 2183	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --
Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

- 1) ☒ Responsive to communication(s) filed on 31 May 2006.
 2a) ☒ This action is **FINAL**. 2b) ☐ This action is non-final.
 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

- 4) ☒ Claim(s) 1-35 is/are pending in the application.
 4a) Of the above claim(s) _____ is/are withdrawn from consideration.
 5) ☐ Claim(s) _____ is/are allowed.
 6) ☒ Claim(s) 1-35 is/are rejected.
 7) ☐ Claim(s) _____ is/are objected to.
 8) ☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

- 9) ☐ The specification is objected to by the Examiner.
 10) ☐ The drawing(s) filed on _____ is/are: a) ☐ accepted or b) ☐ objected to by the Examiner.
 Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).
 Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
 a) ☐ All b) ☐ Some * c) ☐ None of:
 1. ☐ Certified copies of the priority documents have been received.
 2. ☐ Certified copies of the priority documents have been received in Application No. _____.
 3. ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

- | | |
|---|---|
| 1) <input type="checkbox"/> Notice of References Cited (PTO-892) | 4) <input type="checkbox"/> Interview Summary (PTO-413)
Paper No(s)/Mail Date. _____ |
| 2) <input type="checkbox"/> Notice of Draftperson's Patent Drawing Review (PTO-948) | 5) <input type="checkbox"/> Notice of Informal Patent Application (PTO-152) |
| 3) <input checked="" type="checkbox"/> Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)
Paper No(s)/Mail Date <u>3/10/06; 3/19/06</u> | 6) <input type="checkbox"/> Other: _____ |

DETAILED ACTION

1. Claims 1-35 have been examined. Claims 27, 29, 30, 32, 33, and 35 have been amended as per Applicant's request.

Claim Rejections - 35 USC § 102

2. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

3. Claims 1-2, 11-15, 24-26, 32-34 are rejected under 35 U.S.C. 102(b) as being anticipated by Rotenberg et al. (herein Rotenberg).
4. As per Claim 1, Rotenberg teaches: A microprocessor (Abstract), comprising:
an instruction cache configured to store instructions (Section 2.1, first paragraph);
a branch prediction unit (Section 2.1, first paragraph);
a trace cache configured to store a plurality of traces of instructions (Section 2.2, second paragraph); and
a prefetch unit coupled to the instruction cache, the branch prediction unit, and the trace cache (Figure 4, and Section 2.2. The Instruction latch appears to fill the role of a prefetch unit);

wherein the prefetch unit is configured to fetch instructions from the instruction cache until the branch prediction unit outputs a predicted target address (Section 2.2, where it is said that "on a trace cache miss, fetching proceeds normally from the instruction cache); and

wherein if the prefetch unit identifies a match for the predicted target address in the trace cache, the prefetch unit is configured to fetch one or more of the plurality of traces from the trace cache (Section 2.2, where it is said that "on a trace cache hit, an entire trace of instructions is fed into the instruction latch, bypassing the instruction cache).

5. As per Claim 2, Rotenberg teaches: The microprocessor of claim 1, wherein the branch prediction unit is configured to output the predicted target address in response to a prediction that a branch will be taken (Section 2.1, which discloses "They (the BTB banks) serve the role of detecting branches in the instructions currently being fetched and providing their target addresses, in time for the next fetch cycle. A few paragraphs further down, it is stated that this happens "if there is a predicted taken branch").

6. As per Claim 11, Rotenberg teaches: The microprocessor of claim 1, wherein each of the plurality of traces comprises partially-decoded instructions (it is inherent that a trace comprises a partially-decoded instruction, otherwise the necessary control information as show in section 2.2 would not be available).

7. As per Claim 12, Rotenberg teaches: The microprocessor of claim 1, wherein each of the plurality of traces is associated with a tag comprising the address of an earliest instruction, in program order, stored within that trace (Section 2.2, where the tag identifies the starting address of the trace).

8. As per Claim 13, Rotenberg teaches: The microprocessor of claim 1, wherein each of the plurality of traces is associated with a flow control field comprising a label for an instruction to which control will pass for each branch operation comprised in that trace (Section 2.2, the "trace target address" and "trace fall-through address" are both labels which describe where control will flow based on each branch operation, based on the prediction (which is part of another label, "branch flags")).

9. As per Claim 14, Rotenberg teaches: A computer system, comprising:
a system memory (inherent if the system has an instruction cache); and
a microprocessor coupled to the system memory (also inherent in Rotenbergs invention), comprising:
an instruction cache configured to store instructions (Section 2.1, first paragraph);
a branch prediction unit (Section 2.1, first paragraph);
a trace cache configured to store a plurality of traces of instructions (Section 2.2);
and

a prefetch unit coupled to the instruction cache, the branch prediction unit, and the trace cache (Figure 4, and Section 2.2. The Instruction latch appears to fill the role of a prefetch unit);

wherein the prefetch unit is configured to fetch instructions from the instruction cache until the branch prediction unit outputs a predicted target address (Section 2.2, where it is said that "on a trace cache miss, fetching proceeds normally from the instruction cache); and

wherein if the prefetch unit identifies a match for the predicted target address in the trace cache, the prefetch unit is configured to fetch one or more of the plurality of traces from the trace cache (Section 2.2, where it is said that "on a trace cache hit, an entire trace of instructions is fed into the instruction latch, bypassing the instruction cache).

10. As per Claim 15, Rotenberg teaches: The computer system of claim 14, wherein the branch prediction unit is configured to output the predicted target address in response to a prediction that a branch will be taken (Section 2.1, which discloses "They (the BTB banks) serve the role of detecting branches in the instructions currently being fetched and providing their target addresses, in time for the next fetch cycle. A few paragraphs further down, it is stated that this happens "if there is a predicted taken branch").

11. As per Claim 24, Rotenberg teaches: The computer system of claim 14, wherein each of the plurality of traces comprises partially-decoded instructions (it is inherent that a trace comprises a partially-decoded instruction, otherwise the necessary control information as show in section 2.2 would not be available).

12. As per Claim 25, Rotenberg teaches: The computer system of claim 14, wherein each of the plurality of traces is associated with a tag comprising the address of an earliest instruction, in program order, stored within that trace (Section 2.2, where the tag identifies the starting address of the trace).

13. As per Claim 26, Rotenberg teaches: The computer system of claim 14, wherein each of the plurality of traces is associated with a flow control field comprising a label for an instruction to which control will pass for each branch operation comprised in that trace (Section 2.2, the "trace target address" and "trace fall-through address" are both labels which describe where control will flow based on each branch operation, based on the prediction (which is part of another label, "branch flags")).

14. As per Claim 32, Rotenberg teaches: A method, comprising:
fetching instructions from an instruction cache (Section 2.1, paragraphs 1-3);
continuing to fetch instructions from the instruction cache until a branch target address is generated (Section 2.2, where it is said that "on a trace cache miss, fetching proceeds normally from the instruction cache);

if a branch target address is generated, searching trace cache for an entry corresponding to the branch target address (Section 2.2, third paragraph).

15. As per Claim 33, Rotenberg teaches: The method of claim 32, further comprising continuing to fetch instructions from the instruction cache if no entry is identified in the trace cache corresponding to the branch target address (Section 2.2, where it is said that "on a trace cache miss, fetching proceeds normally from the instruction cache).

16. As per Claim 34, Rotenberg teaches: The method of claim 32, further comprising fetching one or more traces from the trace cache if an entry is identified in the trace cache corresponding to the branch target address (Section 2.2, where it is said that "on a trace cache hit, an entire trace of instructions is fed into the instruction latch, bypassing the instruction cache).

Claim Rejections - 35 USC § 103

17. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

18. Claims 3, 16 are rejected under 35 U.S.C. 103(a) as being unpatentable over Rotenberg, in view of Patterson et al. (herein Patterson).

As per Claim 3, Rotenberg teaches the microprocessor of claim 1, but fails to teach: wherein the branch prediction unit is configured to output the predicted target address in response to detection of a branch misprediction. However, Patterson teaches that in order to reduce the penalty for a misprediction, you can fetch both the taken and not taken instructions, and put them in the BTB, which would then be able to immediately output the correct path on a misprediction without having to fetch it (Pages 276-277). While it would increase the cost of the system, the advantage is a smaller misprediction penalty, which may worth the cost, depending on the needs of the system. Therefore, one of ordinary skill in the art at the time the invention was made would have stored both the taken and not taken branch paths in the BTB in order to reduce the misprediction penalty, and thus increasing performance.

19. As per Claim 16, Rotenberg teaches the computer system of claim 14, but fails to teach: wherein the branch prediction unit is configured to output the predicted target address in response to detection of a branch misprediction. However, Patterson teaches that in order to reduce the penalty for a misprediction, you can fetch both the taken and not taken instructions, and put them in the BTB, which would then be able to immediately output the correct path on a misprediction without having to fetch it (Pages 276-277). While it would increase the cost of the system, the advantage is a smaller misprediction penalty, which may worth the cost, depending on the needs of the system. Therefore, one of ordinary skill in the art at the time the invention was made would have stored both the taken and not taken branch paths in the BTB in order to reduce the

misprediction penalty, and thus increasing performance.

20. Claims 4, 10, 17, 23 are rejected under 35 U.S.C. 103(a) as being unpatentable over Rotenberg, in view of Braught.

21. As per Claim 4, Rotenberg teaches: The microprocessor of claim 1, further comprising a trace generator (it is necessary for Rotenbergs invention to have a trace generator in order to create traces), but fails to teach: wherein the trace generator is configured to begin a trace with an instruction corresponding to a label boundary.

Rotenberg teaches starting a trace on a trace cache miss, which then causes the line-fill buffer to begin filling (Section 2.2, fourth paragraph). The trace cache is only checked for a hit in the case of a branch, as the branch predictions determine if it is a hit or not. Therefore, Rotenbergs invention generates a trace when a branch is encountered, but not necessarily on a label boundary. However, Braught teaches that in Assembly Language, most branches operate on labels, which the machine can only interpret when converted to an address (Page 3). In Braughts example on page 3, all branches are labels, making every branch a label boundary. When combined with Rotenbergs invention, this would mean that a trace would be generated with an instruction corresponding to a label boundary, as all the branches would go to labels, and even an address may be interpreted as a label. Given this knowledge, it would have been obvious to one of ordinary skill in the art that Rotenbergs invention begins traces when it

encounters an instruction with a label boundary, as it only begins on branch instructions, which branch to labels.

22. As per Claim 10, Rotenberg teaches: The microprocessor of claim 4, wherein the trace generator is configured to generate traces in response to instructions being decoded (Section 2.2, the trace can not be completed (written into the cache) until the trace target addresses are calculated, which requires the instructions to be decoded).

23. As per Claim 17, Rotenberg teaches: The computer system of claim 14, further comprising a trace generator (it is necessary for Rotenbergs invention to have a trace generator in order to create traces), but fails to teach: wherein the trace generator is configured to begin a trace with an instruction corresponding to a label boundary.

Rotenberg teaches starting a trace on a trace cache miss, which then causes the line-fill buffer to begin filling (Section 2.2, fourth paragraph). The trace cache is only checked for a hit in the case of a branch, as the branch predictions determine if it is a hit or not. Therefore, Rotenbergs invention generates a trace when a branch is encountered, but not necessarily on a label boundary. However, Braught teaches that in Assembly Language, most branches operate on labels, which the machine can only interpret when converted to an address (Page 3). In Braughts example on page 3, all branches are labels, making every branch a label boundary. When combined with Rotenbergs invention, this would mean that a trace would be generated with an instruction corresponding to a label boundary, as all the branches would go to labels, and even an

Art Unit: 2183

address may be interpreted as a label. Given this knowledge, it would have been obvious to one of ordinary skill in the art that Rotenbergs invention begins traces when it encounters an instruction with a label boundary, as it only begins on branch instructions, which branch to labels.

24. As per Claim 23, Rotenberg teaches: The computer system of claim 17, wherein the trace generator is configured to generate traces in response to instructions being decoded (Section 2.2, the trace can not be completed (written into the cache) until the trace target addresses are calculated, which requires the instructions to be decoded).

25. Claims 5-8 and 18-21 are rejected under 35 U.S.C. 103(a) as being unpatentable over Rotenberg and Braught, further in view of Lange et al. (USPN 3,896,419, herein Lange).

26. As per Claim 5, Rotenberg and Braught teach the microprocessor of claim 4, but fail to teach: wherein the trace generator is configured to check the trace cache for a duplicate copy of the trace that the trace generator is constructing. However, Lange teaches a system in which a cache is checked for a match with memory, while the memory is being read at the same time. The advantage of Lange's method is no delay in the overall data fetch cycle if there is no match in the cache, and the memory access can be cancelled before it incurs any system slowdown as well (Column 2, Lines 13-31, and Column 5, Lines 5-10). Given this advantage, it would have been obvious to one of

ordinary skill in the art at the time the invention was made to check the trace cache for a duplicate copy of the trace at the same time the trace was being constructed, in order to minimize or eliminate delay in generating the trace in the case that the trace is not in the cache, and aborting the unnecessary trace if it is found in the cache.

27. As per Claim 6, Lange teaches: The microprocessor of claim 5, wherein if the trace generator identifies a duplicate copy of the trace, the trace generator is configured to discard the trace under construction (Column 5, Lines 5-10).

28. As per Claim 7, Rotenberg teaches: The microprocessor of claim 5, wherein if the trace generator identifies an entry corresponding to a duplicate copy of the trace, the trace generator is configured to check the trace cache for an entry corresponding to a next trace to be generated (Section 2.2. The trace cache is checked every time there is a potential new trace, so when one trace is found and discarded, the next potential new trace will cause the trace cache to be checked again).

29. As per Claim 8, Lange teaches: The microprocessor of claim 7, wherein if the trace generator identifies a trace entry corresponding to the next trace to be generated, the trace generator is configured to discard the trace under construction (Column 5, Lines 5-10).

30. As per Claim 18, Rotenberg and Braught teach the computer system of claim 17, but fail to teach: wherein the trace generator is configured to check the trace cache for a duplicate copy of the trace that the trace generator is constructing. However, Lange teaches a system in which a cache is checked for a match with memory, while the memory is being read at the same time. The advantage of Lange's method is no delay in the overall data fetch cycle if there is no match in the cache, and the memory access can be cancelled before it incurs any system slowdown as well (Column 2, Lines 13-31, and Column 5, Lines 5-10). Given this advantage, it would have been obvious to one of ordinary skill in the art at the time the invention was made to check the trace cache for a duplicate copy of the trace at the same time the trace was being constructed, in order to minimize or eliminate delay in generating the trace in the case that the trace is not in the cache, and aborting the unnecessary trace if it is found in the cache.

31. As per Claim 19, Lange teaches: The computer system of claim 18, wherein if the trace generator identifies a duplicate copy of the trace, the trace generator is configured to discard the trace under construction (Column 5, Lines 5-10).

32. As per Claim 20, Rotenberg teaches: The computer system of claim 18, wherein if the trace generator identifies an entry corresponding to a duplicate copy of the trace, the trace generator is configured to check the trace cache for an entry corresponding to a next trace to be generated (Section 2.2. The trace cache is checked every time there is a potential new trace, so when one trace is found and discarded, the next potential

new trace will cause the trace cache to be checked again).

33. As per Claim 21, Lange teaches: The computer system of claim 20, wherein if the trace generator identifies a trace entry corresponding to the next trace to be generated, the trace generator is configured to discard the trace under construction (Column 5, Lines 5-10).

34. Claims 9 and 22 are rejected under 35 U.S.C. 103(a) as being unpatentable over Rotenberg, in view of Akkary et al. (USPN 6,247,121, herein Akkary).

35. As per Claim 9, Rotenberg teaches the microprocessor of claim 4, but fails to teach: wherein the trace generator is configured to generate traces in response to instructions being retired. While Rotenberg teaches that the instructions need to be decoded (Section 2.2) before the trace can be generated, it is not taught that they need to be retired beforehand. Akkary teaches that instructions are not put into the trace buffers until they have been retired, to ensure that they executed correctly (Column 3, Lines 40-44). Rotenberg discusses in Section 2.3 that some traces are committed but never used, thus evicting a useful trace. By ensuring that the trace is correct, the odds that it will be useful is increased, as an incorrect trace should probably not need to be used. Therefore, one of ordinary skill in the art at the time the invention was made would have waited until an instruction was retired before considering adding it to the trace, in order to ensure accuracy of the trace.

36. As per Claim 22, Rotenberg teaches the computer system of claim 17, but fails to teach: wherein the trace generator is configured to generate traces in response to instructions being retired. While Rotenberg teaches that the instructions need to be decoded (Section 2.2) before the trace can be generated, it is not taught that they need to be retired beforehand. Akkary teaches that instructions are not put into the trace buffers until they have been retired, to ensure that they executed correctly (Column 3, Lines 40-44). Rotenberg discusses in Section 2.3 that some traces are committed but never used, thus evicting a useful trace. By ensuring that the trace is correct, the odds that it will be useful is increased, as an incorrect trace should probably not need to be used. Therefore, one of ordinary skill in the art at the time the invention was made would have waited until an instruction was retired before considering adding it to the trace, in order to ensure accuracy of the trace.

37. Claims 27-31 and 35 are rejected under 35 U.S.C. 103(a) as being unpatentable over Rotenberg, Braught, and Lange, further in view of Akkary.

38. As per Claim 27, Rotenberg teaches: A method, comprising:
starting construction of a new trace if the received instruction is associated with a branch label (Section 2.2. As shown in the rejections for Claims 4 and 17, Rotenberg starts traces on branches, which branch to labels (or addresses, as shown by Braught), thus obviating that all traces are associated with some branch label), but fails to teach:

receiving a retired instruction;

if a previous trace under construction duplicates a trace in a trace cache, delaying construction of the new trace until the received instruction corresponds to a branch label.

While Rotenberg teaches that the instructions need to be decoded (Section 2.2) before the trace can be generated, it is not taught that they need to be retired beforehand. Akkary teaches that instructions are not put into the trace buffers until they have been retired, to ensure that they executed correctly (Column 3, Lines 40-44). Rotenberg discusses in Section 2.3 that some traces are committed but never used, thus evicting a useful trace. By ensuring that the trace is correct, the odds that it will be useful is increased, as an incorrect trace should probably not need to be used. Therefore, one of ordinary skill in the art at the time the invention was made would have waiting until an instruction was retired before considering adding it to the trace, in order to ensure accuracy of the trace.

Lange teaches a system in which a cache is checked for a match with memory, while the memory is being read at the same time. The advantage of Lange's method is no delay in the overall data fetch cycle if there is no match in the cache, and the memory access can be cancelled before it incurs any system slowdown as well (Column 2, Lines 13-31, and Column 5, Lines 5-10). Given this advantage, it would have been obvious to one of ordinary skill in the art at the time the invention was made to check the trace cache for a duplicate copy of the trace at the same time the trace was being constructed, in order to minimize or eliminate delay in generating the trace in the

case that the trace is not in the cache, and aborting the unnecessary trace if it is found in the cache. Therefore, when a trace is found to be in the cache, and the trace generation is aborted, it is obvious that a new trace would not be built until a new branch (which would have to be associated with a branch label) was retired.

39. As per claim 28, Rotenberg teaches: The method of claim 27, further comprising continuing construction of an incomplete trace already in process (Section 2.2).

40. As per Claim 29, Lange teaches: The method of claim 27, further comprising searching the trace cache for duplicate entries (Column 5, Lines 5-10).

41. As per Claim 30, Rotenberg teaches: The method of claim 29, further comprising creating a new entry in the trace cache if no duplicate entry is identified (Section 2.2).

42. As per Claim 31, Lange teaches: The method of claim 29, further comprising discarding a trace if a duplicate entry is identified (Column 5, Lines 5-10).

43. As per Claim 35, Rotenberg teaches: A microprocessor comprising:
means for starting a new trace if the received operation is a first operation at a branch label (Section 2.2. As shown in the rejections for Claims 4 and 17, Rotenberg starts traces on branches, which branch to labels (or addresses, as shown by Braught), thus obviating that all traces are associated with some branch label), but fails to teach:

means for receiving a retired operation;

means for delaying starting a new trace if a previous trace under construction duplicates a trace in trace cache, until the received operation corresponds to a branch label.

While Rotenberg teaches that the instructions need to be decoded (Section 2.2) before the trace can be generated, it is not taught that they need to be retired beforehand. Akkary teaches that instructions are not put into the trace buffers until they have been retired, to ensure that they executed correctly (Column 3, Lines 40-44). Rotenberg discusses in Section 2.3 that some traces are committed but never used, thus evicting a useful trace. By ensuring that the trace is correct, the odds that it will be useful is increased, as an incorrect trace should probably not need to be used. Therefore, one of ordinary skill in the art at the time the invention was made would have waiting until an instruction was retired before considering adding it to the trace, in order to ensure accuracy of the trace.

Lange teaches a system in which a cache is checked for a match with memory, while the memory is being read at the same time. The advantage of Lange's method is no delay in the overall data fetch cycle if there is no match in the cache, and the memory access can be cancelled before it incurs any system slowdown as well (Column 2, Lines 13-31, and Column 5, Lines 5-10). Given this advantage, it would have been obvious to one of ordinary skill in the art at the time the invention was made to check the trace cache for a duplicate copy of the trace at the same time the trace was being constructed, in order to minimize or eliminate delay in generating the trace in the

case that the trace is not in the cache, and aborting the unnecessary trace if it is found in the cache. Therefore, when a trace is found to be in the cache, and the trace generation is aborted, it is obvious that a new trace would not be built until a new branch (which would have to be associated with a branch label) was retired.

Response to Arguments

44. Applicant's arguments with respect to the Section 112, Second Paragraph rejections have been fully considered and are persuasive. The rejection of Claim 28 has been withdrawn.

45. Applicant's arguments filed 5/31/2006 have been fully considered but they are not persuasive. Regarding the Section 102(b) rejections on Claims 1, 14, and some of their dependants, Applicant has argued that the instruction latch does not function as a prefetch unit, that it does not identify a match in the trace cache, fetch instructions from an instruction cache, nor does it fetch from an instruction cache until a branch prediction unit outputs an address, then stop and fetch from the trace cache, and that detecting a trace cache miss is not the same as outputting a predicted target address.

The examiner has interpreted "fetching an instruction" to be the act of providing an instruction to the computer system. Through that definition, the Examiner said that the instruction latch appeared to perform the role of the prefetch unit in Applicant's claims. To clarify Examiner's position, the instruction latch fulfills the role of the prefetch unit, when its inputs are considered as well, which is why the rejection brought in what

Art Unit: 2183

the trace cache and the core fetch unit was doing, as they both fed into the latch, which then finished the fetch operation by providing the appropriate instruction to the system, through either the trace cache or the core fetch unit. Given this clarification, Examiner believes the issues involving finding a match in the trace catch, and fetching instructions from an instruction cache will be resolved.

Regarding the final two points, it is stated in Claim 1 that the prefetch unit fetches from the instruction cache until a branch prediction unit outputs a prediction, and if that prediction is in the trace cache, it fetches from the trace cache. The limitation of "fetches from the instruction cache until the branch prediction unit outputs a predicted target address" had been interpreted to read as when the branch prediction unit outputs a predicted address *not* in the trace cache, as otherwise, the invention would clearly not function as described in the Applicants arguments. Applicant has argued that in the claimed invention, the prefetch unit would stop fetching from the instruction cache as a result of a predicted target address being generated, which would cause the processor to stop running if the value was not in the trace cache, which Examiner interpreted would not be the case, as that would not be an enabled system, rather, Examiner read the last two limitations together, in which the instruction cache is no longer output to the system when there is a trace cache hit from the predicted address, but if there is a predicted address NOT in the trace cache, the processor would continue to run instead of shutting down.

Regarding Claim 32, Applicant has argued that Rotenberg does not teach the claimed invention, as Rotenberg accesses the trace cache in parallel with the instruction

Art Unit: 2183

cache, and does not stop fetching from the instruction cache on a prediction, nor does Rotenberg search the trace cache *only* on a branch target address being generated. Firstly, as explained above, while the two are accessed in parallel, only one can be “fetched”, that is, sent to the system, therefore, if the trace cache generates a hit (does not generate a miss), the instruction cache, while being accessed, will not send instructions to the system, effectively not fetching. As per the second point, the Applicants claim language does not state that the trace cache is searched *only* when a branch target address is generated, just that it is searched when an address is generated. As they operate in parallel, when an address is generated, the trace cache will be examined as well, it just does so in other situations as well, which does not teach away from the claim limitations, as that particular limitation (“only”) is not in the claim. Also, as explained above, Examiner interpreted “until a branch target address is generated” to read on the case the trace cache was a miss as well, to avoid an enablement issue of the processor shutting down, unable to fetch instructions from either source.

46. As per the Section 103(a) rejections, Examiner has corrected the typographical error in the rejection of Claim 27 and the other claims in that section, in which the claim was rejected also in view of Lange.

47. Regarding Claim 27, Applicant has argued that Akkary fails to suggest the limitation in the claim of “receiving a retired instruction”. While Applicant and Examiner

are in agreement that instructions are retired *from* the trace buffers on retirement, the quoted passage of Akkary was intended by the Examiner to show that instructions cannot be shown to have executed correctly until they had been retired, and was not used to show how Akkary's trace buffers functioned. Thus, the Examiner stated that the motivation to combine Akkary with Rotenberg was that one of ordinary skill in the art would recognize the advantage of waiting to add an instruction to a trace until the instruction was guaranteed to have executed correctly, to avoid incorrect traces, as traces are used to increase performance, not degrade it by giving incorrect results. Therefore, Akkary suggests the limitation of using a retired instruction, by providing one of ordinary skill in the art with a motivation for using a retired instruction, as opposed to in-flight instructions.

Regarding the use of Lange, Examiner and Applicant are in complete agreement that Claim 27 does not recite fetching data from a memory, or checking for a cache hit on a memory access. Lange was, however, not used to teach limitations such as those. Instead, Lange was used to teach why it would be advantageous to check for a duplicate copy of a value in a cache, and why you would not want to continue with the operation that is not needed when the value is already present, in Lange's case, it was to prevent an unnecessary memory access that could slow down the system. There is a performance degradation in Rotenberg's system that does come up in this case, as seen in Section 2.3, Point 5. Here, Rotenberg discusses what happens when a trace cache miss occurs when the line-fill buffer is collecting a trace. If this line-fill buffer was collecting a trace which was already in the buffer, then a performance hit would occur,

Art Unit: 2183

as the options are to ignore it, which would remove the performance boost of the cache, to delay servicing it, or to provide multiple buffers, which would create the need for more hardware, which could increase the cost of the system, and also raises the complexity. Given the less complex methods, and that they both involve lowering the efficiency of the processor, being able to keep the line-fill buffer free of unnecessary data clearly provides an increase in performance, and is a problem which Rotenberg has disclosed which Lange provides a solution for, by teaching the idea that a check for duplicate data can end an unnecessary operation.

Conclusion

48. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire **THREE MONTHS** from the mailing date of this action. In the event a first reply is filed within **TWO MONTHS** of the mailing date of this final action and the advisory action is not mailed until after the end of the **THREE-MONTH** shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than **SIX MONTHS** from the mailing date of this final action.


Any inquiry concerning this communication or earlier communications from the examiner should be directed to Robert E. Fennema whose telephone number is (571) 272-2748. The examiner can normally be reached on Monday-Friday, 8:00-5:30.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Eddie Chan can be reached on (571) 272-4162. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Robert E Fennema
Examiner
Art Unit 2183

RF


RICHARD L. ELLIS
PRIMARY EXAMINER